Identity Access Management – IAM

There are 2 types of Access available for IAM users

1. Management Console Access Type
2. Programmatic Access Type

Management Console Access Type

- Allows users to sign in via the AWS Management Console using a username and password.
- You can enable MFA (Multi-Factor Authentication) for added security.
- Typically assigned to administrators, developers, or users who need a visual interface to manage AWS services.

Programmatic Access Type:

- Grants access via **AWS CLI, SDKs, and API calls** using an **Access Key (Access Key ID & Secret Access Key)**.
- Used for automation, applications, and scripts that interact with AWS.
- No console access—only API-level interactions.

Account Alias:

An **AWS Account Alias** is a user-friendly name that replaces the default **AWS account ID** in the sign-in URL, making it easier to remember and share.

- Makes it easier to remember and share the login URL
- Enhances security by avoiding exposure of the account ID
- Useful for organizations managing multiple AWS accounts

Let's Divide the Lab session into 2 parts:

1. Lab Tasks by assigning the user with Management console access type
2. Lab Tasks by assigning the user with Programmatic access type

Management Console Access Type

1. Create an IAM User **"user0"** with all default options
2. Note down the IAM user Sign-In url
3. Login as "user0" and observe that the user has No permissions
4. Set an **account alias** for the AWS Account as **"CloudLearn"**
5. Use the alias to login as **"user0"**
6. Create an IAM user **"user1"** with Full access on the entire AWS Account by attaching an IAM policy "**Administrator Access**"

There are 3 different ways that we can provide permissions to the users:

- Attach an existing Policy
- Add users to a Group
- Copy from existing IAM User

7. Login as **"User1"** and verify the Administrator access Permissions
    a. Check that the user1 can see all the list of users
    b. Create a new IAM user as **"user1"**
    c. Navigate to EC2 service, create a security Group
8. Create an IAM user **"user2"** with read-only permission on IAM Service only By attaching the policy directly to the user
9. Login as **"user2"** and verify the read-only permissions on IAM Service only
    a. User2 can perform Read action on IAM Service
    b. User2 cannot Read any other service (ex: EC2, S3 etc..)
10. Create a new IAM Policy **"EC2FullAccess"**(Custom Policy), which can provide Full access on EC2 service.
11. Create a Group "**EC2Admins**" and attach the above custom Policy "**EC2FullAccess"**
12. Create a user "**user3**" and add the user into the group "**EC2Admins"**
13. Login as **"User3"** and observe that he hot full permissions on EC2
14. Assign additional Permissions **"Read-only Access"** to the user **"user3"** and Observe that he got the below access permissions
    a. Full Access on EC2 service
    b. Read-Only Access on all other AWS Services

Note: IAM fully supports **Deny policies** to explicitly restrict access to AWS resources. **Explicit Deny** takes precedence over Allow, making it a powerful security mechanism.

15. Create a Deny Policy **"IAMFullAccess-Deny",** which can deny all actions on IAM Service
16. Attach the Deny Policy **"IAMFullAccess-Deny",** to the existing user **"user1"** who already has Full access
17. Login as "user1" and observe that he cannot perform any action on IAM Servcie.